Menggambar **LINGKARAN** (1/7)

- Persamaan umum LINGKARAN :

$$(x - x_0)^2 + (y - y_0)^2 = r^2$$

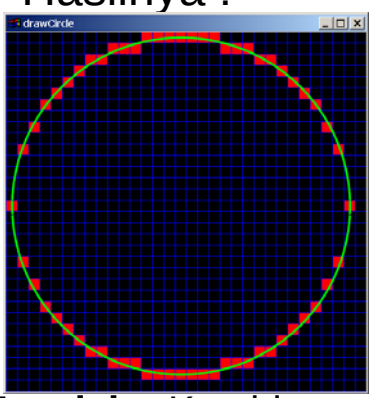
dengan

$$y = y_0 \pm \sqrt{r^2 - (x - x_0)^2}$$

Contoh program menggambar lingkaran :

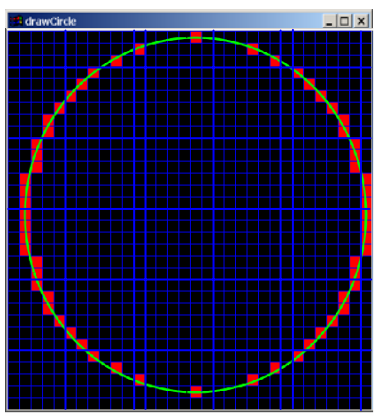
```
void circleSimple(int xCenter, int yCenter, int radius,
    Color c) {
    int x, y, r2;

    r2 = radius * radius;
    for (x = -radius; x <= radius; x++) {
       y = (int)(sqrt(r2 - x*x) + 0.5);
       setPixel(xCenter + x, yCenter + y, c);
       setPixel(xCenter + x, yCenter – y, c);
    }
}
```

Hasilnya :



Masalah : Kemiringan tangen pada suatu titik > 1, x tidak bekerja.

Bagaimana bila dilakukan terhadap y ?



Solusi : gabungkan keduanya sehingga didapat lingkaran simetris untuk x dan y.

Cara ini disebut

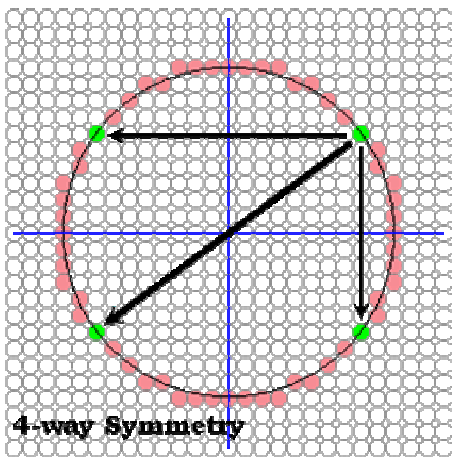
2-way symmetry

Menggambar **LINGKARAN** (2/7)

- **4-way symmetry**

```
void circle4Way(int xCenter, int yCenter, int radius,
   Color c) {
   int x, y, r2;

   setPixel(xCenter, yCenter + radius, c);
   setPixel(xCenter, yCenter – radius, c);

   r2 = radius * radius;
   for (x = 1; x <= radius; x++) {
      y = (int)(sqrt(r2 - x*x) + 0.5);
      setPixel(xCenter + x, yCenter + y, c);
      setPixel(xCenter + x, yCenter – y, c);
      setPixel(xCenter - x, yCenter + y, c);
      setPixel(xCenter - x, yCenter – y, c);
   }
}
```

Hasil :



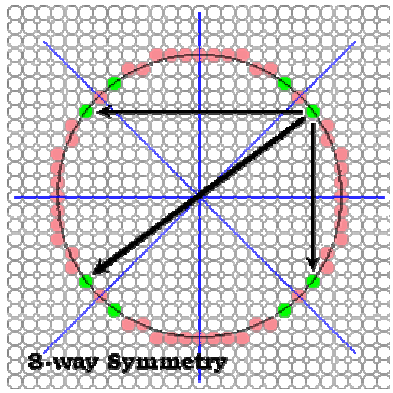
Lebih cepat dari yang pertama, tapi tidak lebih baik dalam menampilkan lingkaran

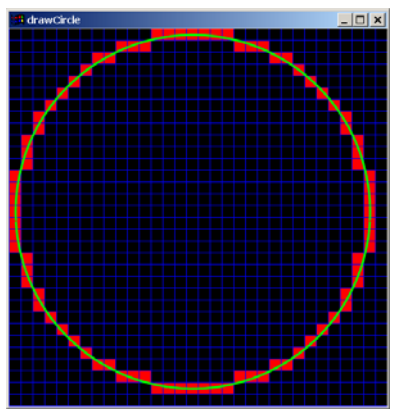
Menggambar **LINGKARAN** (3/7)

- **8-way symmetry**

```
void circle8Way(int xCenter, int yCenter, int radius, Color c) {
    int x, y, r2;

  setPixel(xCenter, yCenter + radius, c);
  setPixel(xCenter, yCenter – radius, c);
  setPixel(xCenter + radius, yCenter, c);
  setPixel(xCenter - radius, yCenter, c);

  r2 = radius * radius;
  x = 1;
  y = (int)(sqrt(r2 – 1) + 0.5);
  while (x < y) {
    setPixel(xCenter + x, yCenter + y, c);
    setPixel(xCenter + x, yCenter – y, c);
    setPixel(xCenter - x, yCenter + y, c);
    setPixel(xCenter - x, yCenter – y, c);
    setPixel(xCenter + y, yCenter + x, c);
    setPixel(xCenter + y, yCenter – x, c);
    setPixel(xCenter - y, yCenter + x, c);
    setPixel(xCenter - y, yCenter – x, c);

    x += 1;
    y = (int)(sqrt(r2 – x*x) + 0.5);
  }
  if (x == y) {
    setPixel(xCenter + x, yCenter + y, c);
    setPixel(xCenter + x, yCenter – y, c);
    setPixel(xCenter - x, yCenter + y, c);
    setPixel(xCenter - x, yCenter – y, c);
  }
}
```

Hasil :





Dengan memanfaatkan diagonal garis yang melewati pusat lingkaran, kita dapat menyusuri x yang koordinatnya telah diubah (pertukaran x dan y) secara serempak, sehingga y menjadi bagian dari lingkaran

Menggambar **LINGKARAN** (4/7)

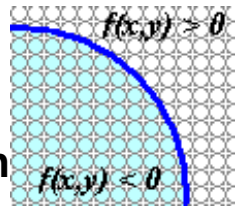
- **Fungsi Disciminator**

  Telah diketahui bahwa : $(x - x_0)^2 + (y - y_0)^2 = r^2$

  dan dapat ditulis sebagai suatu fungsi : **$f(x,y) = x^2 + y^2 - r^2$**

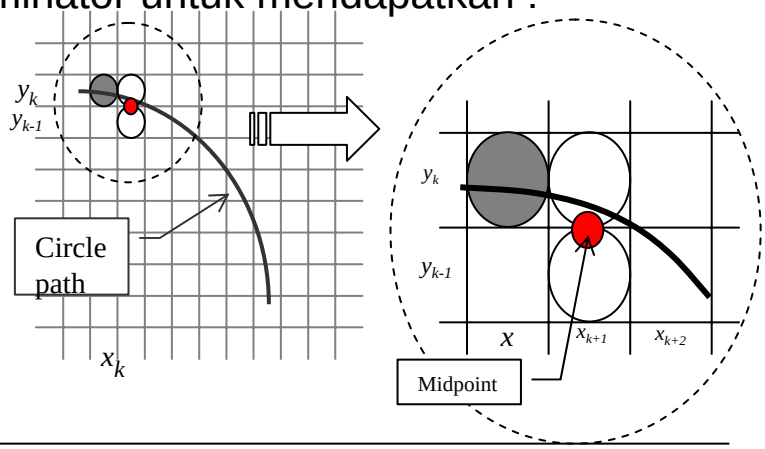
  Fungsi Discriminator :

  f(x,y) < 0 untuk titik di dalam lingkaran
  f(x,y) > 0 untuk titik di luar lingkaran
  f(x,y) = 0 untuk titik yang terletak pada lingkaran



- **Algoritma Titik Tengah Lingkaran (Midpoint Circle Algorithm)**
  - Bila diketahui suatu titik : (x_k,y_k), maka titik berikutnya apakah di (x_k+1, y_k), or (x_k+1, y_k-1) ?
  - Misal titik tengahnya (midpoint) : $(x_k+1, y_k) = 0.5$
  - Gunakan fungsi discriminator untuk mendapatkan :

    $f(x,y) = x^2 + y^2 - r^2$



Menggambar **LINGKARAN** (5/7)

- Algoritma Titik Tengah Lingkaran ……..*(lanjutan)*
  - Dengan menggunakan *midpoint* di antara 2 kandidat pixel, kita dapat mencari *Parameter Keputusan,* P_k, untuk mendapatkan plot pixel berikutnya :

$$P_k = f(x_k + 1, y_k - ½)$$
$$= (x_k + 1)^2 + (y_k - ½)^2 - r^2$$

P_k : **- ve** , titik tengah berada di dalam lingkaran,

plot = (x_k+1 , y_k),

Update *P* : $f(x+1, y) = (x + 1)^2 + y^2 - r^2$

$f(x+1, y) = (x^2 + 2x + 1) + y^2 - r^2$

$f(x+1, y) = f(x, y) + 2x + 1$

P_{k+1} P_k

Inkremen : $P + = 2x + 1$

+ ve , titik tengah berada di luar lingkaran,

plot = (x_k+1 , y_k-1)

Update *P* : $f(x+1, y-1) = (x + 1)^2 + (y-1)^2 - r^2$

$f(x+1, y-1) = (x^2 + 2x + 1) + (y^2 - 2y+2-r^2)$

$f(x+1, y-1) = f(x, y) + 2x + 2y - 1$

P_{k+1} P_k

Inkremen : $P + = 2x - 2y + 2$

Menggambar **LINGKARAN** (6/7)

- Kita dapat memiliki beragam titik awal lingkaran, namun diasumsikan inkremen dimulai dari (0,r), sehingga P_0 dapat dihitung :

$$p_0 = f(1, r–0.5) = 1^2 + (r – 0.5)^2 – r^2$$

$$p_0 = f(1, r–0.5) = 1 + (r^2 – r + 0.25) – r^2$$

$$p_0 = 1.25 – r$$

- **Algoritma Titik Tengah (MidPoint) selengkapnya :**
  1. Input radius, *r*, and titik tengah lingkaran (x_c, y_c). Titik awal di-plot pada (0, *r*) – yang merupakan titik tengah lingkaran asli,
  2. Hitung nilai awal *Parameter Keputusan* : $p_0 = \frac{5}{4} - r$
  3. Pada x_k, dimulai dengan $k = 0$, uji nilai p_k :

     Jika $p_k < 0$, maka titik selanjutnya (x_{k+1}, y_k) dan $p_{k+1} = p_k + 2x_{k+1} + 1,$

     Untuk hal lain, titik berikutnya (x_{k+1}, y_k-1) dan $p_{k+1} = p_k + 2x_{k+1} + 1 - 2y_{k+1}.$
  4. Tentukan titik simetri pada 7 octant lainnya.
  5. Ambil titik aktual untuk titik tengah lingkaran pada (x_c, y_c) dimana ($x + x_c$, $y + y_c$).
  6. Ulangi langkah 3 sampai 5 hingga tercapai $x \geq y$.

Menggambar **LINGKARAN** (7/7)

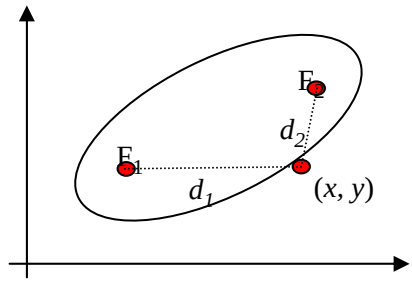
```
void circleMidpoint(int xCenter, int yCenter, int radius, Color c) {
   int x = 0;
   int y = radius;
   int p = (5 - radius*4)/4;

   circlePoints(xCenter, yCenter, x, y, c);
   while (x < y) {
      x++;
      if (p < 0) {
         p += 2*x+1;
      } else {
         p += 2*(x-y+1);
         y--;
      }
      circlePoints(xCenter, yCenter, x, y, c);
   }
}

void circlePoints(int cx, int cy, int x, int y, Color c) {
   if (x == 0) {
      setPixel(cx, cy + y, c);
      setPixel(cx, cy – y, c);
      setPixel(cx + y, cy, c);
      setPixel(cx - y, cy, c);
   } else if (x == y) {
      setPixel(cx + x, cy + y, c);
      setPixel(cx - x, cy + y, c);
      setPixel(cx + x, cy – y, c);
      setPixel(cx - x, cy – y, c);
   } else if (x < y) {
      setPixel(cx + x, cy + y, c);
      setPixel(cx - x, cy + y, c);
      setPixel(cx + x, cy – y, c);
      setPixel(cx - x, cy – y, c);
      setPixel(cx + y, cy + x, c);
      setPixel(cx - y, cy + x, c);
      setPixel(cx + y, cy – x, c);
      setPixel(cx - y, cy – x, c);
   }
}
```

Menggambar **ELLIPS** (1/4)

- Persamaan umum ellips :

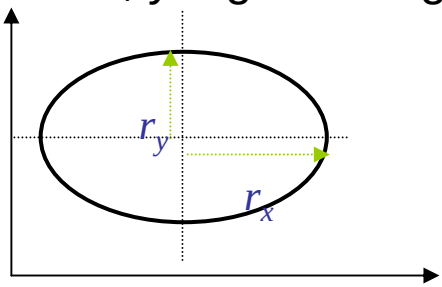
  $d_1 + d_2 = konstan$

  atau



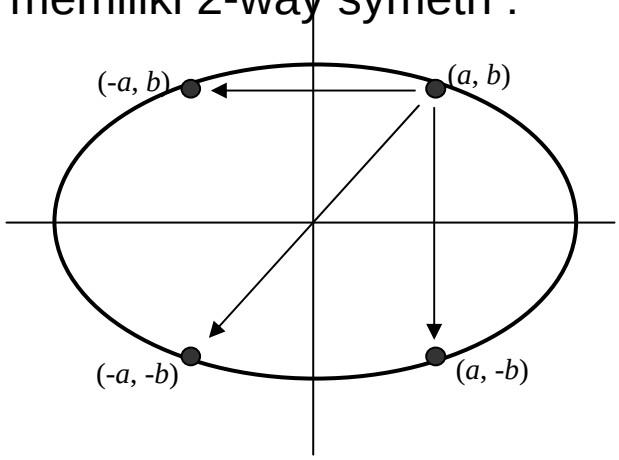
$$\sqrt{(x-x_1)^2+(y-y_1)^2}+\sqrt{(x-x_2)^2+(y-y_2)^2}=\text{constant}$$

- Namun, yang akan digunakan adalah **ellips standard** :



$$\left(\frac{x-x_c}{r_x}\right)^2+\left(\frac{y-y_c}{r_y}\right)^2=1$$

- Ellips hanya memiliki 2-way symetri :



Menggambar **ELLIPS** (2/4)

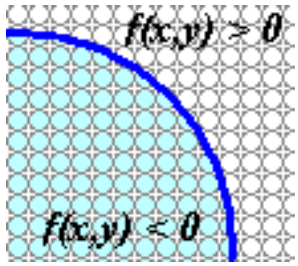
- Membangun ellips :
  - Pusat ellips standard : $\left(\frac{x}{r_x}\right)^2 + \left(\frac{y}{r_y}\right)^2 = 1$
  - Fungsi Discriminator : $f_e(x,y) = r_y^2 x^2 + r_x^2 y^2 - r_x^2 r_y^2$

    dimana :

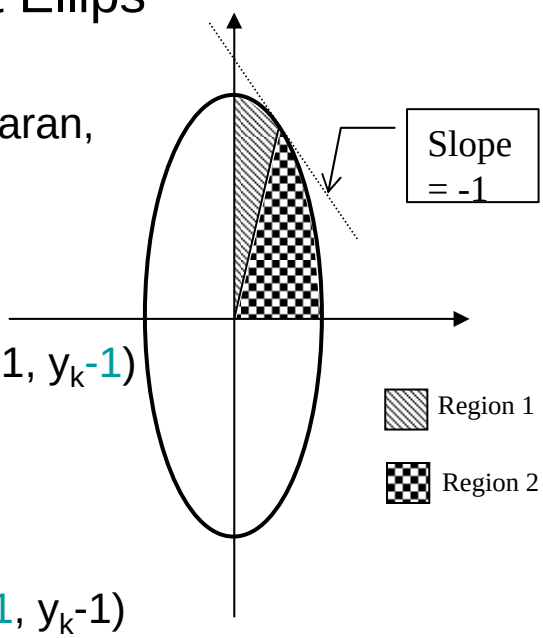
    $f_e(x,y) < 0$ untuk suatu titik di dalam ellips

    $f_e(x,y) > 0$ untuk suatu titik di luar ellips

    $f_e(x,y) = 0$ untuk suatu titik pada ellips



- Algoritma Titik Tengah/MidPoint Ellips
  - Ellips berbeda dengan lingkaran
  - Pendekatannya sama dengan lingkaran, tapi berbeda dalam sampling arah.
  - Region 1 :
    - Sampling arah x
    - Pilihannya antara (x_k+1, y_k), or (x_k+1, y_k-1)
    - Midpoint: $(x_k+1, y_k-0.5)$
  - Region 2 :
    - Sampling arah y
    - Pilihannya antara (x_k, y_k-1), or (x_k+1, y_k-1)
    - Midpoint: $(x_k+0.5, y_k-1)$



Menggambar **ELLIPS** (3/4)

- Parameter Keputusan
  - Region 1: $p1_k = f_e\left(x_k + 1, y_k - \tfrac{1}{2}\right)$

    $p1_k$ –ve:
    - midpoint di dalam
    - pilih pixel (x_{k+1}, y_k)

    $p1_k$ +ve:
    - midpoint di luar
    - pilih pixel (x_{k+1}, y_k-1)

  - Region 2 : $p2_k = f_e\left(x_k + \tfrac{1}{2}, y_k - 1\right)$

    $p2_k$ –ve:
    - midpoint di dalam
    - pilih pixel (*xk*+1, *yk*-1)

    $p2_k$ +ve:
    - midpoint di di luar
    - pilih pixel (*xk*, *yk*-1)

Menggambar **ELLIPS** (4/4)

- **Algoritma Titik Tengah/MidPoint Ellips**

1. Input r_x, r_y titik tengah ellips (x_c, y_c). Titik awal sama degnan pusat ellips asli (0, r_y).
2. Nilai awal parameter keputusan pada region 1: $p1_0 = r_y^2 - r_x^2 r_y + \frac{1}{4} r_x^2$
3. Untuk setiap x_k pada region 1, dimulai dari k = 0, uji $p1_k$:

   Jika $p1_k < 0$, titik berikutnya (x_k+1, y_k) dan

   $$p1_{k+1} = p1_k + 2r_y^2 x_{k+1} + r_y^2,$$

   Keadaan lain, titik berikutnya (x_{k+1}, y_k-1) dan

   $$p1_{k+1} = p1_k + 2r_y^2 x_{k+1} - 2r_x^2 y_{k+1} + r_y^2.$$
4. Tentukan titik simetri pada 3 octant lainnya
5. Ambil titik aktual untuk pusat ellips (x_c, y_c) dimana ($x + x_c$, $y + y_c$).
6. Ulangi langkah 3 - 6 hinga tercapai $2r_y^2 x \geq 2r_x^2 y$i.
7. Nilai awal parameter keputusan region 2:

   $$p2_0 = r_y^2 \left(x_0 + \tfrac{1}{2}\right)^2 + r_x^2 \left(y_0 - 1\right)^2 - r_x^2 r_y^2$$

8. Untuk setiap y_k pada 2, dimulai dari k = 0, uji $p2_k$:

   Jika $p2_k > 0$, titik berikutnya (x_k, y_k-1) dan

   Keadaan lain, titik berikutnya (x_k+1, y_k-1) dan

9. Tentukan titik simetri pada 3 octant lainnya
10. Ambil titik aktual untuk pusal ellips (x_c, y_c) dimana ($x + x_c$, $y + y_c$).
11. Ulangi langkah 8 - 10 hingga $y < 0$.